Developers Guide for Installing and Building Cubit Windows 2000 / Visual .NET Version 0.9.4

This guide has been developed to help Cubit developers correctly set up and build the Cubit executable, including the Claro GUI (Claro) . We assume that serious Cubit developers will need to work with both GUI and command line versions of Cubit.

This guide assumes that you already have the operating system (Windows 2000 or XP) installed on your machine. You may also have some of the software described here already installed on your system. If so, read through the applicable section to verify that the installed software is a correct, recent version, and that any settings and directory structures described are set up correctly.

There are several places in the instructions where you will see the comment **NOTE** in bold letters. Please pay attention to this information as it may save you much time and trouble in your installation.

Assumptions

The document assumes that the user has sufficient skill with the Windows operating system to perform the following tasks:

1. Set system variables. Cubit installation requires setting of system variables such as CUBITROOT. System variables can be set in Windows 2000 by right-clicking the “my computer” icon, and selecting properties from the popup menu. On the system properties panel, select the Advanced tab, and then push the “Environment Variables” button in the middle of the panel. In the “Environment Variables” panel, use the bottom half titled “System Variables”. Use the New, Edit, and Delete buttons as appropriate to set the correct system variables. **REMEMBER: Applications will need to be closed and restarted to reflect the changes made to system variables.**
2. Download and install software from the internet or from CD. All required software is listed below.

Prerequisites

Before downloading and building Cubit, you will need two accounts and several software tools.

Malla Account

In order to download the software, you will need an account on malla.sandia.gov which contains the cubit repository. Contact Bob Kerr (rakerr@sandia.gov) to get an account and password.

Developer Password to Cubit website

In order to obtain access to developer only tools, libraries, etc., you will need to obtain a password to the developer area of the cubit website (http://cubit.sandia.gov/training). Contact Bob Kerr (rakerr@sandia.gov) to get an account and password to this developer site.

Software

Install the following software, if the correct version is not already loaded on your machine:

- WinZip compression/decompression utility version 8.0 or later. A demo version may be downloaded from www.winzip.org .

- CCVSSH is a secure shell program that allows secure connection to Sandia computers such as cubit.sandia.gov. CCVSSH can be obtained from the cubit developers website (http://cubit.sandia.gov/release/developers_tools/WINDOWS). Dowload file ccvssh_install.exe. You will need your developer area password to access this area.

- WinCVS **version 1.20** (NOTE: the latest version 1.30 is not compatible with CCVSSH, so make sure and get version 1.20 for now. WinCVS may be downloaded from www.wincvs.org.

- CMAKE is an open source, cross-platform build system used to build Cubit. CMAKE can be downloaded from www.cmake.org. Install version 2.0.5 or later.

- C++ compiler. You will need to obtain and download Visual C++ .NET for compiling the source code. Ask your supervisor how to obtain a copy of this program, and install using the default setup. **NOTE**: Use care if the older Visual C++ 6.0 is installed on the same machine. Depending on the PATH variable, one or both could get incorrect run-time libraries and work incorrectly.

Where to Put the Code

You will be downloading two main code groups. The first is the cubit source code. You will select a directory for this, and the code will be created in a sub-folder called cubit under your folder. For example, if you choose a project directory called c:\cubit_project, the cubit source tree will begin at c:\cubit_project\cubit. In the remainder of the document, we will be referring to this location as *CUBIT_ROOT.* When path or file information is called for, simply substitute in the directory (e.g., c:\cubit_project\cubit) for the *CUBIT_ROOT* portion of the path.

The second code group downloaded for windows are the cubit libraries and third party files. Again, select a top level directory, and the code will be put into a directory called windows_libs_net under this directory. Thus, if you choose c:\cubit_project as the top level, CVS will create a directory c:\cubit_project\windows_libs_net to put the code in. In the remainder of the document, we will be referring to this location as *CUBIT_LIB.* When path or file information is called for, simply substitute in the directory (e.g., c:\cubit_project\windows_libs_net) for the *CUBIT_LIB* portion of the path.

Downloading Cubit Source

If you are lucky, you will now be able to start downloading Cubit source code, provided you are careful in setting up CVS correctly.

- **Define system variable CVSROOT**. The value should be **:ext:*user*@malla.sandia.gov:/usr/local/eng_sci/CVS** where the *user* is the user name supplied for you malla account.

- **Run the CVSSSH login program once**. From your desktop, select start->programs->CCVSSH->ccvssh login. When prompted, enter your Malla password.

- **Setup WinCVS.** Run WinCVS, either from the icon on your desktop, or by selecting start->programs->WinCVS->WinCVS. In the WinCVS GUI, select admin->preferences. In the preferences dialog, under the General tab, enter the CVSROOT value exactly as you did above for the system variable. In the Authentication pulldown, select SSH server. Click the Globals tab. Check the box labeled “Use TCP/IP compression”, and put a 9 in the text box. Click the Ports tab. At the bottom, check the box labeled “Check for an alternate rsh name:” and in the text field, enter the full path to the CCVSSH executable (for the default installation, this is “c:\Program Files\CCVSSH\ccvssh.exe” without the quotes). Click okay. If all has gone well, you can now begin downloading the Cubit source code.

- **Checkout the cubit code using WinCVS.** In the WinCvs GUI, select create->checkout module. In the checkout settings dialog under "Enter the module name and path on the server:" type cubit (caps are significant). Under "Local folder to checkout to:", enter the name of the directory you wish to have your cubit source under. CVS will create a folder called cubit under the folder you select in this step. You may put the code anywhere. Click okay to begin the download. This will take some time to complete. Below, you may wish to write the value of *CUBIT_ROOT* to be used later (for example, c:\cubit_project\cubit):

  CUBIT_ROOT = _____

- **Checkout the windows library code using WinCVS.** In the WinCvs GUI, select create->checkout module. In the checkout settings dialog under "Enter the module name and path on the server:" type windows_libs_net (caps are significant). Under "Local folder to checkout to:", enter the name of the directory you wish to have your cubit libraries under. CVS will create a folder called windows_libs_net under the folder you select in this step. You may put the code anywhere; we recommend putting it in the same folder as the cubit code checked out above. In our examples we will use the local folder name c:\cubit_project. Click okay to begin the download. This will take some time to complete. Below, you may wish to write the value of *CUBIT_LIB* to be used later (for example, c:\cubit_project\windows_libs_net):

  CUBIT_LIB = _____

- **Define the correct path to the latest current cubit directories and libraries**. To make this simple and less prone to error, a vbscript file has been provided to assist. The script is located in the *CUBIT_LIB*\bin directory. Double click on the set_cubit_path.vbs icon to run the program. This will create a system variable named CUBIT_PATH which contains a rather long specification of the paths to the various cubit components and libraries.

- **Update the system variable PATH**. Edit this variable, and (if it is not already there) add the text "%CUBIT_PATH%;" at the end of the PATH variable. This will add all of the needed path information for Windows to find Cubit libraries and components to run correctly. **Note:** If you have previously run cubit using the current machine, you may have old path information in your PATH variable. Edit the variable to remove references to cubit-specific code (e.g., paths which refer to verdict, camal, acis, mesquite, etc). Old paths in your PATH variable may cubit to build and/or run incorrectly.

- **Define the system variable CUBITROOT**. Its value should be the full path to the windows_libs_net directory created in the last step (for example, c:\cubit_project\windows_libs_net).

Building Cubit

- **Setup Visual C++ .NET.** Start up Visual C++ .NET. From the menu, select tools->options. In the left-hand pane, click on Projects->VC++ Directories. Under the "Show directories for:" pulldown, select executables. Scroll to the bottom of the Directories box, and click on the empty box outline at the bottom of the list. Enter the path *CUBIT_LIB*\bin. Select OK to finish. You may now close Visual C++ .NET if you wish.

Building The Cubit Command Line Version

- **Build the .NET solution files.** Run CmakeSetup from the icon on your screen. Under "Where is the source code:", Enter *CUBIT_ROOT*. Under "Where to build the binaries:", we strongly suggest you enter a different build directory (e.g., c:\cubit_project\build\cubit). Below, this location is written as *BUILD_DIR.* In the Build For: menu, select the compiler (Visual 6 or Visual 7 .NET). Press the configure button. CMakeSetup runs its configuration step. After the first configure step, there may be variables which are still unknown. These will be highlighted in red on the left panel of the CmakeSetup screen. Unless you wish to modify CmakeSetup from the defaults, you can usually just press configure until all variables default. When there are no red variables left, press the OK button to write out all CMakeSetup information. **NOTE:** if you have the cygwin version of bison and flex installed on your machine, you will need to modify two variables: set BISON_EXE variable to point to your cygwin bison.exe (usually some path of the form c:*your_cygwin_area*\cygwin\bin\bison.exe). Similarly, set the FLEX_EXE variable to point to your cygwin version of flex.exe. Press configure, then OK.

- **Build the Cubit Code.** Open Visual C++ .NET. Under the file menu, select Open->Project. Select the solution file *BUILD_DIR*\cubit.sln **.** Select Build->Build Solution, and wait for the project to fully build. This may take some time.

- **Setup Visual .NET to run the proper executable.** In the solution explorer pane, right-click on the ALL_BUILD project and select

Properties… On the Property Pages pop-up in the left-hand pane, select Configuration Properties->Debugging. Under the Action->Command argument, enter *BUILD_DIR*\debug\cubitx.exe (for debug) or *BUILD_DIR*\release\cubitx.exe. Select OK.

- **Run Cubit.** To run the finished executable, press F5, or select run from the menu buttons.

- **NOTE**: if windows cannot run the executable because some .dll or .lib cannot be found, it is because the PATH variable did not get set correctly. Try restarting windows, which will fully redefine the PATH variable. If this does not work, you may need to contact cubit-dev@sandia.gov or your supervisor to get it defined correctly.

Building Cubit GUI Version

- **Build the .NET solution files**. Run CMakeSetup. Under "Where is the source code:", Enter *CUBIT_ROOT*. Under "Where to build the binaries:", we strongly suggest you enter a different build directory (e.g., c:\cubit_project\build\cubit). Below, this location is written as *BUILD_DIR.* In the "Build For:" menu, select the compiler (Visual 7 .NET). Find the BUILD_INTERFACE Cache Value, and turn it ON. Press Configure. Two more variables may appear in Red on the left side of the Cache Values. Set COPY_INTERFACE to ON. Set the CUBITI_COPY_DIR value to *CUB IT_LIB* /claro/component/cubit. Press Configure. CMakeSetup runs its configuration step. If the step is fully successful, the OK button will become active. Press the OK button to write out all CMakeSetup information. Sometimes, the configuration takes two or more iterations to fully resolve the build parameters. Keep pressing configure until all red variables have turned grey, then press OK. **NOTE:** if you have the cygwin version of bison and flex installed on your machine, you will need to modify two variables: set BISON_EXE variable to point to your cygwin bison.exe (usually some path of the form c:*your_cygwin_area*\cygwin\bin\bison.exe). Similarly, set the FLEX_EXE variable to point to your cygwin version of flex.exe. Press configure, then OK.

- **Build the Cubit Code.** Open Visual C++ .NET. Under the file menu, select Open->Project. Select the path *BUILD_DIR*\cubit.sln Select Build->Build Solution, and wait for the project to fully build. This may take some time. Running the project now will run the GUI and bring up Cubit within the GUI. You may change any of the Cubit code and rebuild to see the code changes in action. You may not change the GUI in any way using this configuration. Those few who actually have a license to QT and are

working on the GUI will have a separate, custom setup of CMakeSetup which will create a new Claro GUI when changes are made.

- **Setup Visual .NET to run the proper executable** In the solution explorer pane (upper right), right-click on the ALL_BUILD project and select Properties… On the Property Pages pop-up in the left-hand pane, select Configuration Properties->Debugging. Under the Action->Command argument, enter *CUBIT_LIB*\claro\bin\clarox_d.exe (for debug) or *CUBIT_LIB*\claro\bin\clarox.exe. Select OK.

- **Run Cubit** To run the finished executable, press F5, or select run from the menu buttons.

- NOTE: if windows cannot run the executable because some .dll or .lib cannot be found, it is because the PATH variable did not get set correctly. Try restarting windows, which will fully redefine the PATH variable. If this does not work, you may need to contact cubit-dev@sandia.gov or your supervisor to get it defined correctly.